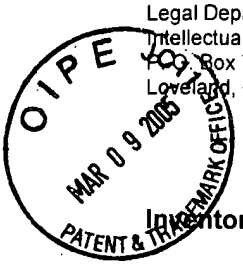
AGILENT TECHNOLOGIES, INC.
Legal Department, DL429
Intellectual Property Administration
P.O. Box 7599
Loveland, Colorado 80537-0599

ATTORNEY DOCKET NO. 10992461-1



IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor(s): Johl et al.

Serial No.: 09/494,817 Examiner: Kenneth Tang

Filing Date: January 31, 2000 Group Art Unit: 2127

Title: METHOD AND SYSTEM FOR INCREASING PERFORMANCE BY SUBSTITUTING FINITE STATE MACHINE CONTROL WITH HARDWAR-IMPLEMENTED DATA STRUCTURE MANIPULATION

COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria VA 22313-1450

TRANSMITTAL OF APPEAL BRIEF

Sir:

Transmitted herewith is the Appeal Brief in this application with respect to the Notice of Appeal filed on

The fee for filing this Appeal Brief is (37 CFR 1.17(c)) **$500.00**.

(complete (a) or (b) as applicable)

The proceedings herein are for a patent application and the provisions of 37 CFR 1.136(a) apply.

☐ (a) Applicant petitions for an extension of time under 37 CFR 1.136 (fees: 37 CFR 1.17(a)(1)-(5)) for the total number of months checked below:

☐	one month	$ 120.00
☐	two months	$ 450.00
☐	three months	$1020.00
☐	four months	$1590.00

☐ The extension fee has already been filled in this application.

☒ (b) Applicant believes that no extension of term is required. However, this conditional petition is being made to provide for the possibility that applicant has inadvertently overlooked the need for a petition and fee for extension of time.

Please charge to Deposit Account **50-1078** the sum of $500.00. At any time during the pendency of this application, please charge any fees required or credit any overpayment to Deposit Account **50-1078** pursuant to 37 CFR 1.25.

A duplicate copy of this transmittal letter is enclosed.

☒ I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

Date of Deposit: March 4, 2005 **OR**

☐ I hereby certify that this paper is being facsimile transmitted to the Patent and Trademark Office on the date shown below.

Date of Facsimile:

Typed Name: Kevin D. Jablonski

Signature: _____

Respectfully submitted,

Johl et al.

By _____

Kevin D. Jablonski
Attorney/Agent for Applicant(s)

Reg. No. 50,401

Date: March 4, 2005

Telephone No. 425-822-3668

PATENT
ATTORNEY DOCKET NO. 10992461-1

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

IN RE APPLICATION OF:Johl et al.
APPLICATION NO.: 09/494,817
FILED: January 31, 2000
FOR: Method and System for Increasing Performance by Substituting Finite State Machine Control with Hardware-Implemented Data Structure Manipulation

EXAMINER: K. Tang
ART UNIT: 2127
CONF. NO: 3999

Appeal Brief

Commissioner for Patents
Alexandria, VA 22313-1450

BRIEF OF APPELLANTS

Appellants Johl et al. have previously filed a Notice of Appeal from the action of the Examiner in finally rejecting all of the claims that were considered in this application. This Brief is being filed under the provisions of 37 C.F.R. § 1.192. The Filing Fee corresponding to this Appeal Brief, as set forth in 37 C.F.R. § 1.17(c), is to be deducted from the Deposit Account identified in the accompanying transmittal.

03/10/2005 HALI11 00000033 501078 09494817
01 FC:1402 500.00 DA

REAL PARTY IN INTEREST

The real party in interest comprises Agilent Technologies, Inc., by way of assignment from Manraj Singh Johl, Joseph H. Steinmetz, and Matthew P. Wakeley ("Johl et al.") who are the named inventors and are captioned in the present brief.

RELATED APPEALS AND INTERFERENCES

None.

STATUS OF CLAIMS

Claims 1-20 are pending. Claims 1-6 and 8-18 are appealed.

STATUS OF AMENDMENTS

No amendment has been filed subsequent to the final rejection.

SUMMARY OF INVENTION

The present invention is directed to system and method for implementing control logic using data structure management, as opposed to finite state machines, to manipulate outbound data streams in a fibre channel. As such, a fibre channel interface according to the present invention converts data streams from one data structure to another based on a particular operation to be performed. Typically, several different operations and/or data manipulations are required to convert outbound data streams into a suitable format for a communication bus. The manner in which the fibre channel manipulates the data stream is one aspect of the present invention that is novel.

It is well known in the art that fibre channels are used as a communications interface between a computer-level bus architecture and a data transmission-level network. When such a fibre channel is used, data streams that are received from

the network typically comprise a serial communication. The fibre channel is able to convert the received serial data stream into a parallel data stream that is suitable for transmission onto a computer bus, such as a PCI bus. Likewise, when data streams are to be sent from the fibre channel to a remote location on the network, the parallel data stream from the computer bus is converted to a serial data stream suitable for transmission on the network. In the past, this conversion and manipulation of outbound data sequences has been performed by devices such as fibre channel interfaces using complex finite state machines within an interface controller. A good example of such a conventional fibre channel interface is the subject matter of the teachings of the cited and applied reference, U.S. Patent No. 5,809,328 to Nogales.

Embodiments of the present invention, however, are not directed to a simple fibre channel interface controller having finite state machines as used in the past and as disclosed in Nogales. Rather, a fibre channel interface according to the present invention utilizes a fibre channel controller having a number of subcontrollers, called managers, to distinguish between different "contexts" in which data may be received by the fibre channel interface. A context describes a task or thread that is processed by the controller and is a fundamental unit of task management for the controller. For example, in one embodiment, six different contexts are associated with six different managers, which represent six different transistions that outbound data may undergo (transmission, completion, non-fabric determination, etc). See specification, page 22, line 21 to page 23, line 25. Thus, each manager is provided with a particular data structure to store individual contexts. This relationship can be likened somewhat to classes (managers having data structures) and instances (data in contexts) in object-oriented programming.

When data (typically in a format called outbound descriptor block (ODB)) is received as part of a data sequence at the outbound sequence manager (OSM) of the fibre channel interface, the appropriate manager is called upon to process the data in context according to its particular data structure. Then, when the data is required for a different purpose, such as transmission to a remote node on the network, the data (now stored in a first data structure) may be transferred to a different context (*i.e.* a second data structure) by simple data manipulation as

opposed to running the data through a complex finite state machines as has been done in the past.

Furthermore, each of the tasks may be performed concurrently (each on a different ODB, of course) during the same clock cycle. Thus, instead of stepping a single ODB (and consequently, each bit in each ODB) through numerous finite state machines (each taking a single clock cycle), numerous ODBs may be concurrently processed in different stages associated with different managers representing different contexts in which each ODB may possibly be.

THE ISSUES

In the Advisory action mailed December 15, 2004, U.S. Patent No. 5,815,649 to Utter was presented in lieu of an "Official Notice" and held forth as a refutation of the applicant's challenge of the original utilization of the Official Notice. The Advisory action further claimed that applicants did not raise an objection or argument to the Examiner's use of Official Notice until after the final Office action. Notwithstanding the fact that neither Office action in this pending case specifically cited the Examiner's use of an "Official Notice," applicants clearly addressed the obviousness rejections and presented arguments that refute the Examiner's interpretation of what is and what is not well known in the industry, regardless of whether the term "Official Notice" was used.

As such, applicants are now confused then as to whether the rejections as they stood in the final Office action stand or if the Utter reference is being used in some capacity in lieu of "Official Notice" in a new set of rejections. However, in the interests of avoiding further delays in the prosecution of this case, the applicants have addressed both scenarios for the purposes of this Appeal Brief.

Thus, the issues in the appeal are as follows:

1. Whether Nogales (5,809,328), whether considered alone or in conjunction with Utter (5,815,649), teaches or suggests all of the features and recitations as respectively recited by claims 1-6.

2. Whether Nogales (5,809,328), whether considered alone or in conjunction with Utter (5,815,649), teaches or suggests all of the features and recitations as respectively recited by claims 8-12.

3. Whether Nogales (5,809,328), whether considered alone or in conjunction with Utter (5,815,649), teaches or suggests all of the features and recitations as respectively recited by claims 13-18.

GROUPING OF CLAIMS

The applicant suggests the following grouping of Claims according to the Issues noted above:

I. Issue 1 is directed toward pending claims 1-6 such that these claims stand or fall together with respect to this issue.

II. Issue 2 is directed toward pending claim 8-12 such that these claims stand or fall together with respect to this issue.

III. Issue 3 is directed toward pending claims 13-18 such that these claims stand or fall together with respect to this issue.

ARGUMENT

Issue 1: Nogales, whether considered alone or in conjunction with Utter, fails to teach or suggest all of the recitations as respectively recited by claims 1-6.

A1. As to claims 1-6, claims 2-6 depend either directly or indirectly from independent claim 1. Thus, claims 2-6 include all of the features and limitations of claim 1, in combination with its own respective features and limitations.

Claims 1-6 have been rejected under 35 U.S.C. § 103(a) as being unpatentable over Nogales (5,809,328). Regarding the independent claim, claim 1 recites: A method for implementing a hardware controller that concurrently executes a number of tasks by carrying out operations on behalf of the tasks, the method comprising determining a format for a context, comprising stored information related to a task, that represents the task, determining possible states, and transitions between states, that a context representing a task currently executed by the hardware controller can occupy at each point in the execution of the task, transitions representing operations performed on behalf of a task by the hardware controller, partitioning the states and operations carried out by the hardware controller into a number of managers each containing a number of related states and carrying out a number of operations associating each manager with a data structure for storing contexts occupying states contained by the manager, defining a data-structure-manipulator manager that implements the data structures and that transfers contexts from one data structure to another, defining a command interface to the data-structure-manipulator manager for each manager, and implementing the managers and data-structure-manipulator manager, according to the determined states and transitions, so that, when a first manager carries out an operation that results in transition of a context to a state contained in a second manager, the first manager generates a command to the data-structure-manipulator manager to transfer the context from the data structure associated with the first manager to the data structure associated with the second manager.

Thus, under pending claim 1 (and dependent claims 2-6), the method recited is directed toward manipulating outbound descriptor blocks (ODBs) during a communication session using a fibre controller. More specifically, each context that an ODB may currently be in is associated with a context, which is, in turn, associated with a manager for handling ODBs in a particular state that are to transition to another state. Thus, for every possible transition in which an ODB may undergo, the system includes a dedicated manager having an associated data structure operable to store the contexts of the particular ODBs that are set to transition from one particular state to another. For example, a manager exists for transitioning an ODB

from a fabric state to a non-fabric state, a manager exists for transitioning an ODB from an idle state to a complete state, a manager exists for transitioning an ODB from an idle state to a transmit state, etc. Furthermore, each manager is managed by a data-structure manipulator manager such that when a first manager carries out an operation that results in transition of a context to a state contained in a second manager, the first manager generates a command to the data-structure-manipulator manager to transfer the context from the data structure associated with the first manager to the data structure associated with the second manager. As a result, several transitions between several different states may be concurrently executed and controlled by the data-structure-manipulator manager.

A2. The primary cited and applied reference, Nogales (5,809,328), merely teaches a known method which encompasses the very problems that the present invention seeks to ameliorate. In an unrelated and not as detailed approach, Nogales teaches, generally, an apparatus for adapting transmissions between an industry-standard data bus of a host computer and a fibre channel coupled between the host computer and a peripheral storage sub-system. See generally, abstract of Nogales. The apparatus is aimed at sharing data between a cluster of work stations in an effort to utilize several processors for parallel computing when dealing with bottleneck communication issues. As such, the system and method taught by Nogales utilizes a fibre channel controller (31) and a main processor (22) of the host computer to control the manipulation of data in shared registers in buffer memory (30) throughout a cluster of work stations. See column 6, lines 42-57 of Nogales. Essentially, the host processor determines the routing of operations (read or write) to be performed and decides which of the work stations or peripheral devices in the cluster will perform the operation. The host processor uses fibre channel links to each peripheral device for assigning operations to be performed remotely. See generally, FIG. 1 and column 2, lines 54-62 of Nogales.

Referring to FIG. 2 and column 6, lines 58-67 of Nogales, in the case of a write command, the fibre channel controller reads the data from the buffer memory (30) and sends the data to a gigabit link module (GLM, 32) for processing from one data structure (PCI bus architecture) to a another data structure (serial bus or

network architecture). Likewise, in the case of a read command, the fibre channel controller reads the data received on the fibre channel at the GLM (32) and stores the received data (after a parallel-to-serial conversion in the buffer memory (30)) until directed elsewhere later by the host processor. As a result, data may be transferred back and forth via a fibre link to peripheral devices.

Nogales, however, is completely silent as to how the data stored in the buffer memory is converted from one data format (serial) to another (parallel) when outbound data is sent to the GLM or received from the GLM. Furthermore, Nogales simply does not teach transferring the data from one data structure to another as the fibre channel controller and the processor share one data structure in the buffer memory. The designation of serial or parallel is a data format as opposed to a data structure. See column 6, lines 43-45 and column 6 lines 60-62 of Nogales. So even though, the Office action correctly noted that the GLM block is configured to perform a serial-to-parallel conversion for write commands and a parallel-to-serial conversion for read commands, Nogales provides no more detail as to how this conversion is accomplished.

Furthermore, Nogales certainly does not rise to the level of description that may encompass the recitations of claim 1. Nogales does not show any cognition, let alone teach a particular manner in which parallel to serial conversions of data take place. Simply stating that a conversion takes place (which is not novel to the invention is Nogales) does not teach the conversion process. Nogales, by conventional systems and known methods, uses one or more finite state machines within the GLM to accomplish the conversion from one data format to the next as is the case with the above-mentioned prior art; one of the very problems that that the present invention is aimed at solving. Thus, there are several key differences in the teachings of Nogales and the present invention as detailed below.

A3. To establish *prima facie* obviousness of a claimed invention, all of the claim recitations must be taught or suggested by the prior art; (*In re Royka*, 490 F.2d 981, 180 USPQ 580 (CCPA 1974)), and "all words in a claim must be considered in judging the patentability of that claim against the prior art;" (*In re Wilson*, 424 F.2d 1382, 1385, 165 USPQ 494, 496 (CCPA 1970)). Further, if prior art, in any material

respect teaches away from the claimed invention, the art cannot be used to support an obviousness rejection. *In re Geisler*, 116 F.3d 1465, 1471, 43 USPQ2d 1362, 1366 (Fed Cir. 1997). Moreover, if a modification would render a reference unsatisfactory for its intended purpose, the suggested modification / combination is impermissible. See MPEP § 2143.01

A4 Nogales, whether considered alone or in conjunction with Utter or what was known in the art at the time of the invention, does not teach or suggest determining possible states, and transitions between states of outbound descriptor blocks. Claim 1 recites determining possible states, and transitions between states, that a context representing a task currently executed by the hardware controller can occupy at each point in the execution of the task, transitions representing operations performed on behalf of a task by the hardware controller.

The Office action dated September 8, 2004 cited PCI Interface Logic in column 6, line 34 of Nogales as teaching this concept. See Office action, page 2, section 4. In this citation, Nogales teaches a logic block (PCI Interface Logic (28)) that facilitates the movement of command control blocks (CCBs) from one memory (namely, host memory (13) to another memory (namely, buffer memory (30).

This concept of moving CCBs (which are not an outbound descriptor blocks (ODBs)) between memories is not the same as manipulating an ODB from one data structure to another. Even if one were to assume that these are related concepts, simply citing "PCI Interface Logic" still does not teach *determining* possible states, and transitions between states, that a context representing a task currently executed by the hardware controller can occupy at each point in the execution of the task, transitions representing operations performed on behalf of a task by the hardware controller. Clearly, Nogales shows no understanding or appreciation of determining states and transitions between states that a context representing a task currently executed by the hardware controller can occupy. At best, Nogales teaches PCI Interface Logic that helps facilitate communication between a PCI bus, i.e., host memory and buffer memory by already knowing what transitions are to take place by defined logic. There is certainly no disclosure in Nogales as to how this communication is carried out. It is quite a stretch to argue that determining

communication possible patterns involving contexts of a task having states and transitions between states wherein the states are associated with dedicated managers is taught by the word "logic" when used in conjunction with a fibre channel controller.

A5. Nogales, whether considered alone or in conjunction with Utter or any other known art at the time of the invention, does not teach or suggest partitioning states and operations into managers. Claim 1 recites partitioning the states and operations carried out by the hardware controller into a number of managers each containing a number of related states and carrying out a number of operations.

The Office action admits, "Nogales fails to explicitly teach having a number of managers containing related/common states." See the Office action dated September 8, 2004, section 5, page 3. This admission is directed to the failure of Nogalas to teach the recitation of partitioning the states and operations carried out by the hardware controller into a number of managers each containing a number of related states and carrying out a number of operations. The Office action dated September 8, 2004 originally stated that this recitation was well known in the art at the time of the invention. In an Advisory action dated December 15, 2004, the Examiner cited Utter as an example of this known art. See section 10 of the Advisory action citing column 7, lines 50-63 of Utter. Simply put Utter does not cure the deficiency admitted by the Office action.

The cited and applied section of Utter is directed to a configuration manager for managing different partitions of distributed memory. In this context, the word "partition" means the actual portion of memory that is dedicated to a single identity. Thus, a block of memory, such as a computer hard drive, may contain a number of partitions, such as C:\, D:\, and Z:\. Quite differently, claim 1 recites partitioning the states and operations carried out by the hardware controller into a number of managers. In this context, the word partitioning does not refer to a portion of memory but rather is used as a verb in that the states and operations are partitioned (*e.g.*, divided, distributed among, spread out, *etc.*) between managers. The Utter reference is wholly unrelated to the present invention and wholly unrelated to Nogales.

Furthermore, Nogales falls significantly short of teaching all of the recitations of claim 1. In essence, the Examiner holds the position that Nogales, by disclosing that the GLM block performs parallel-to-serial conversions or vice versa and that grouping together similar tasks as a generally accepted and known programming tenet renders obvious each and every recitation in claim 1. Such broad, conclusory statements do not come close to adequately addressing the issue of motivation to combine, are not evidence of obviousness, and therefore are improper as a matter of law. *In re Dembiczak*, 175 F.3d 994, 999, 50 USPQ2d 1614, 1617 (Fed. Cir. 1999).

A6. Nogales, whether considered alone or in conjunction with Utter or any other known art at the time of the invention, does not teach or suggest associating each manager with a data structure for storing contexts occupying states contained by the manager. Claim 1 recites associating each manager with a data structure for storing contexts occupying states contained by the manager.

The Office action dated September 8, 2004 contends that Nogales teaches this recitation at column 5, lines 7-13. In this cited and applied section of Nogales, it is disclosed that the FC controller (31) and the processor (22) shares data structures and data buffers that are maintained in the buffer memory (30). That is, Nogales teaches that a single type of data structure is shared between the processor and the FC controller.

Claim 1, however, recites associating each manager with a data structure for storing contexts occupying states contained by the manager. That is, each manager is associated with its own dedicated type of data structure such that when a first manager carries out an operation that results in transition of a context to a state contained in a second manager, the first manager generates a command to the data-structure-manipulator manager to transfer the context from the data structure associated with the first manager to the data structure associated with the second manager. Once again, Nogales shows no cognition of the concept of states or transitions between states, thus, Nogales cannot possibly teach associating managers with a data structure for storing contexts, let alone the specific data

structures that correspond to particular states in which the contexts may be, which are also, in turn, associated with the managers.

A7. In view of the foregoing, the Appellants respectfully submit that the rejections of claims 1-6 must fail for impropriety, and that Board must accordingly overturn these rejections.

Issue 2: Nogales, whether considered alone or in conjunction with Utter fails to teach or suggest all of the recitations as respectively recited by claims 8-12.

B1. As to claims 8-12, claims 9-12 depend either directly or indirectly from independent claim 8. Thus, claims 9-12 include all of the features and limitations of claim 1, in combination with its own respective features and limitations.

Claims 8-12 have been rejected under 35 U.S.C. § 103(a) as being unpatentable over Nogales (5,809,328). Regarding the independent claim, claim 8 recites: A method for implementing a hardware controller that concurrently executes a number of tasks, the method comprising: representing each task executed by the hardware controller as a context, each context occupying a state determined by the contents of at least one field within the context, a context transitioning from one state to another state when the hardware controller carries out an operation on behalf of the task represented by the context; partitioning hardware controller operations and associated context states into a number of logical managers; associating each logical manager with one of a number logical data structures for storing contexts occupying states within the logical manager; and implementing the logical managers and a data-structure manipulator that contains the contexts, logical data structures, and a command interface through which each logical manager issues commands to direct the data-structure manipulator to transfer a context from the data structure associated with the logical manager to a different data structure.

Thus, under pending claim 8 (and dependent claims 9-12), the method recited is directed again toward manipulating outbound descriptor blocks (ODBs) during a communication session using a fibre controller. More specifically, each

context that an ODB may currently be in is associated with a context, which is, in turn, associated with a manager for handling ODBs in a particular state that are to transition to another state. Thus, for every possible transition in which an ODB may undergo, a dedicated manager having an associated data structure operable to store the contexts of the particular ODBs that are set to transition from one particular state to another. For example, a manager exists for transitioning an ODB from a fabric state to a non-fabric state, a manager exists for transitioning an ODB from an idle state to a complete state, a manager exists for transitioning an ODB from an idle state to a transmit state, etc. Furthermore, each manager is managed by a data-structure manipulator manager such that when a first manager carries out an operation that results in transition of a context to a state contained in a second manager, the first manager generates a command to the data-structure-manipulator manager to transfer the context from the data structure associated with the first manager to the data structure associated with the second manager. As a result, several transitions between several different states may be concurrently executed and controlled by the data-structure-manipulator manager.

B2. The primary cited and applied reference, Nogales (5,809,328), merely teaches a known method which encompasses the very problems that the present invention seeks to ameliorate. In an unrelated and not as detailed approach, Nogales teaches, generally, an apparatus for adapting transmissions between an industry-standard data bus of a host computer and a fibre channel coupled between the host computer and a peripheral storage sub-system. See generally, abstract of Nogales. The apparatus is aimed at sharing data between a cluster of work stations in an effort to utilize several processors for parallel computing when dealing with bottleneck communication issues. As such, the system and method taught by Nogales utilizes a fibre channel controller (31) and a main processor (22) of the host computer to control the manipulation of data in shared registers in buffer memory (30) throughout a cluster of work stations. See column 6, lines 42-57 of Nogales. Essentially, the host processor determines the routing of operations (read or write) to be performed and decides which of the work stations or peripheral devices in the cluster will perform the operation. The host processor uses fibre channel links to

each peripheral device for assigning operations to be performed remotely. See generally, FIG. 1 and column 2, lines 54-62 of Nogales.

Referring to FIG. 2 and column 6, lines 58-67 of Nogales, in the case of a write command, the fibre channel controller reads the data from the buffer memory (30) and sends the data to a gigabit link module (GLM, 32) for processing from one data structure (PCI bus architecture) to a another data structure (serial bus or network architecture). Likewise, in the case of a read command, the fibre channel controller reads the data received on the fibre channel at the GLM (32) and stores the received data (after a parallel-to-serial conversion in the buffer memory (30)) until directed elsewhere later by the host processor. As a result, data may be transferred back and forth via a fibre link to peripheral devices.

Nogales, however, is completely silent as to how the data stored in the buffer memory is converted from one data format (serial) to another (parallel) when outbound data is sent to the GLM or received from the GLM. Furthermore, Nogales simply does not teach transferring the data from one data structure to another as the fibre channel controller and the processor share one data structure in the buffer memory. The designation of serial or parallel is a data format as opposed to a data structure. See column 6, lines 43-45 and column 6 lines 60-62 of Nogales. So even though, the Office action correctly noted that the GLM block is configured to perform a serial-to-parallel conversion for write commands and a parallel-to-serial conversion for read commands, Nogales provides no more detail as to how this conversion is accomplished.

Furthermore, Nogales certainly does not rise to the level of description that may encompass the recitations of claim 8. Nogales does not show any cognition, let alone teach a particular manner in which parallel to serial conversions of data take place. Simply stating that a conversion takes place (which is not novel to the invention is Nogales) does not teach the conversion process. Nogales, by conventional systems and known methods, uses one or more finite state machines within the GLM to accomplish the conversion from one data format to the next as is the case with the above-mentioned prior art; one of the very problems that that the

present invention is aimed at solving. Thus, there are several key differences in the teachings of Nogales and the present invention as detailed below.

B3. To establish *prima facie* obviousness of a claimed invention, all of the claim recitations must be taught or suggested by the prior art; (*In re Royka*, 490 F.2d 981, 180 USPQ 580 (CCPA 1974)), and "all words in a claim must be considered in judging the patentability of that claim against the prior art;" (*In re Wilson*, 424 F.2d 1382, 1385, 165 USPQ 494, 496 (CCPA 1970)). Further, if prior art, in any material respect teaches away from the claimed invention, the art cannot be used to support an obviousness rejection. *In re Geisler*, 116 F.3d 1465, 1471, 43 USPQ2d 1362, 1366 (Fed Cir. 1997). Moreover, if a modification would render a reference unsatisfactory for its intended purpose, the suggested modification / combination is impermissible. See MPEP § 2143.01

B4 Nogales, whether considered alone or in conjunction with Utter or what was known in the art at the time of the invention, does not teach or suggest representing each task executed by a hardware controller as a context that may occupy a state determined by the contents of at least one field within the context. Claim 8 recites representing each task executed by the hardware controller as a context, each context occupying a state determined by the contents of at least one field within the context.

The Office action dated September 8, 2004 cited the same reasoning as cited with regard to the rejection of claim 1. Presumably then, the Office action contends then that the disclosure of PCI Interface Logic (column 6, line 34 of Nogales) teaches this concept. See Office action, page 2, section 4. In this citation, Nogales teaches a logic block (PCI Interface Logic (28)) that facilitates the movement of command control blocks (CCBs) from one memory (namely, host memory (13) to another memory (namely, buffer memory (30).

This concept of moving CCBs (which are not an outbound descriptor blocks (ODBs)) between memories is not the same as manipulating an ODB from one data structure to another. Even if one were to assume that these are related concepts, simply citing "PCI Interface Logic" still does not teach *representing* each task executed by the hardware controller as a context, each context occupying a state

determined by the contents of at least one field within the context. Clearly, Nogales shows no understanding or appreciation of representing states and transitions between states that a context representing a task currently executed by the hardware controller can occupy. At best, Nogales teaches PCI Interface Logic that helps facilitate communication between a PCI bus, *i.e.*, host memory and buffer memory by already knowing what transitions are to take place by defined logic. There is certainly no disclosure in Nogales as to how this communication is carried out. It is quite a stretch to argue that determining communication possible patterns involving contexts of a task having states and transitions between states wherein the states are associated with dedicated managers is taught by the word "logic" when used in conjunction with a fibre channel controller.

B5. Nogales, whether considered alone or in conjunction with Utter or any other known art at the time of the invention, does not teach or suggest partitioning hardware controller managers. Claim 8 recites partitioning hardware controller operations and associated context states into a number of logical managers.

The Office action admits, "Nogales fails to explicitly teach having a number of managers containing related/common states." See the Office action dated September 8, 2004, section 5, page 3. This admission is presumably directed to the failure of Nogalas to teach the recitation of partitioning hardware controller operations and associated context states into a number of logical managers. The Office action dated September 8, 2004 originally stated that this recitation was well known in the art at the time of the invention. In an Advisory action dated December 15, 2004, the Examiner cited Utter as an example of this known art. See section 10 of the Advisory action citing column 7, lines 50-63 of Utter. Simply put Utter does not cure the deficiency admitted by the Office action.

The cited and applied section of Utter is directed to a configuration manager for managing different partitions of distributed memory. In this context, the word "partition" means the actual portion of memory that is dedicated to a single identity. Thus, a block of memory, such as a computer hard drive, may contain a number of partitions, such as C:\, D:\, and Z:\. Quite differently, claim 8 recites partitioning hardware controller operations and associated context states into a number of

logical managers. In this context, the word partitioning does not refer to a portion of memory but rather is used as a verb in that the states and operations are partitioned (*e.g.*, divided, distributed among, spread out, *etc.*) between managers. The Utter reference is wholly unrelated to the present invention and wholly unrelated to Nogales.

Furthermore, Nogales falls significantly short of teaching all of the recitations of claim 8. In essence, the Examiner holds the position that Nogales, by disclosing that the GLM block performs parallel-to-serial conversions or vice versa and that grouping together similar tasks as a generally accepted and known programming tenet renders obvious each and every recitation in claim 8. Such broad, conclusory statements do not come close to adequately addressing the issue of motivation to combine, are not evidence of obviousness, and therefore are improper as a matter of law. *In re Dembiczak*, 175 F.3d 994, 999, 50 USPQ2d 1614, 1617 (Fed. Cir. 1999).

B6. Nogales, whether considered alone or in conjunction with Utter or any other known art at the time of the invention, does not teach or suggest associating each logical manager with one of a number of logical data structures. Claim 8 recites associating each logical manager with one of a number of logical data structures for storing contexts occupying states within the logical manager.

The Office action dated September 8, 2004 contends that Nogales teaches this recitation at column 5, lines 7-13 as presumed by the similarity to claim 1 and the rejection based on the reasoning of claim 1. In this cited and applied section of Nogales, it is disclosed that the FC controller (31) and the processor (22) shares data structures and data buffers that are maintained in the buffer memory (30). That is, Nogales teaches that a single type of data structure is shared between the processor and the FC controller.

Claim 8, however, recites associating each logical manager with a data logical structure for storing contexts occupying states contained by the logical manager. That is, each logical manager is associated with its own dedicated type of logical data structure such that when a first manager carries out an operation that results in transition of a context to a state contained in a second manager, the first manager

generates a command to the data-structure-manipulator manager to transfer the context from the data structure associated with the first manager to the data structure associated with the second manager. Once again, Nogales shows no cognition of the concept of states or transitions between states, thus, Nogales cannot possibly teach associating logical managers with a dedicated logical data structure for storing contexts, let alone the specific data structures that correspond to particular states in which the contexts may be, which are also, in turn, associated with the logical managers.

B7. In view of the foregoing, the Appellants respectfully submit that the rejections of claims 8-12 must fail for impropriety, and that Board must accordingly overturn these rejections.

Issue 3: Nogales, whether considered alone or in conjunction with Utter, fails to teach or suggest all of the recitations as respectively recited by claims 13-18.

C1. As to claims 13-18, claims 14-18 depend either directly or indirectly from independent claim 13. Thus, claims 14-18 include all of the features and limitations of claim 13, in combination with its own respective features and limitations.

Claims 13-18 have been rejected under 35 U.S.C. § 103(a) as being unpatentable over Nogales (5,809,328). Regarding the independent claim, claim 13 recites: A subcomponent controller within a communication controller comprising: data storage elements that together compose a number of contexts for storing information related to a sequence of data to be exchanged through a communications medium connected to the communication controller; logical managers that are each associated with a data structure and that each carries out operations on behalf of contexts stored within the associated data structure; and a data-structure manipulator that implements a number of data structures for storing contexts and that transfer contexts between data structures in response to receiving transfer commands from the logical managers.

Thus, under pending claim 13 (and dependent claims 14-18), the system recited is directed toward a controller for manipulating outbound descriptor blocks (ODBs) during a communication session using a fibre controller. More specifically, each state that an ODB may currently be in is associated with a data storage element that together comprise a number of contexts, which are, in turn, associated with a number of logical managers for handling sequences of data in a particular state that are to transition to another state. Thus, for every possible context in which data may exist, a dedicated logical manager having an associated data structure operable to store the contexts of the particular data that are set to transition from one particular state to another. For example, a logical manager exists for transitioning an ODB from a fabric state to a non-fabric state, a manager exists for transitioning an ODB from an idle state to a complete state, a manager exists for transitioning an ODB from an idle state to a transmit state, etc.

Furthermore, each logical manager is managed by a data-structure manipulator such that when a first manager carries out an operation that results in transition of a context to a state contained in a second manager, the first manager generates a command to the data-structure-manipulator to transfer the context from the data structure associated with the first logical manager to the data structure associated with the second logical manager. As a result, several transitions between several different states may be concurrently executed and controlled by the data-structure-manipulator.

C2. Again, the primary cited and applied reference, Nogales (5,809,328), merely teaches a known method which encompasses the very problems that the present invention seeks to ameliorate. In an unrelated and not as detailed approach, Nogales teaches, generally, an apparatus for adapting transmissions between an industry-standard data bus of a host computer and a fibre channel coupled between the host computer and a peripheral storage sub-system. See generally, the abstract of Nogales. The apparatus is aimed at sharing data between a cluster of work stations in an effort to utilize several processors for parallel computing when dealing with bottleneck communication issues. As such, the system and method taught by Nogales utilizes a fibre channel controller (31) and a

main processor (22) of the host computer to control the manipulation of data in shared registers in buffer memory (30) throughout a cluster of work stations. See column 6, lines 42-57 of Nogales. Essentially, the host processor determines the routing of operations (read or write) to be performed and decides which of the work stations or peripheral devices in the cluster will perform the operation. The host processor uses fibre channel links to each peripheral device for assigning operations to be performed remotely. See generally, FIG. 1 and column 2, lines 54-62 of Nogales.

Referring to FIG. 2 and column 6, lines 58-67 of Nogales, in the case of a write command, the fibre channel controller reads the data from the buffer memory (30) and sends the data to a gigabit link module (GLM, 32) for processing from one data structure (PCI bus architecture) to a another data structure (serial bus or network architecture). Likewise, in the case of a read command, the fibre channel controller reads the data received on the fibre channel at the GLM (32) and stores the received data (after a parallel-to-serial conversion in the buffer memory (30)) until directed elsewhere later by the host processor. As a result, data may be transferred back and forth via a fibre link to peripheral devices.

Nogales, however, is completely silent as to how the data stored in the buffer memory is converted from one data format (serial) to another (parallel) when outbound data is sent to the GLM or received from the GLM. Furthermore, Nogales simply does not teach transferring the data from one data structure to another as the fibre channel controller and the processor share one data structure in the buffer memory. The designation of serial or parallel is a data format as opposed to a data structure. See column 6, lines 43-45 and column 6 lines 60-62 of Nogales. So even though, the Office action correctly noted that the GLM block is configured to perform a serial-to-parallel conversion for write commands and a parallel-to-serial conversion for read commands, Nogales provides no more detail as to how this conversion is accomplished.

Furthermore, Nogales certainly does not rise to the level of description that may encompass the recitations of claim 13. Nogales does not show any cognition, let alone teach a particular manner in which parallel to serial conversions of data

take place. Simply stating that a conversion takes place (which is not novel to the invention is Nogales) does not teach the conversion process. Nogales, by conventional systems and known methods, uses one or more finite state machines within the GLM to accomplish the conversion from one data format to the next as is the case with the above-mentioned prior art; one of the very problems that that the present invention is aimed at solving. Thus, there are several key differences in the teachings of Nogales and the present invention as detailed below.

C3. To establish *prima facie* obviousness of a claimed invention, all of the claim recitations must be taught or suggested by the prior art; (*In re Royka*, 490 F.2d 981, 180 USPQ 580 (CCPA 1974)), and "all words in a claim must be considered in judging the patentability of that claim against the prior art;" (*In re Wilson*, 424 F.2d 1382, 1385, 165 USPQ 494, 496 (CCPA 1970)). Further, if prior art, in any material respect teaches away from the claimed invention, the art cannot be used to support an obviousness rejection. *In re Geisler*, 116 F.3d 1465, 1471, 43 USPQ2d 1362, 1366 (Fed Cir. 1997). Moreover, if a modification would render a reference unsatisfactory for its intended purpose, the suggested modification / combination is impermissible. See MPEP § 2143.01

C4 Nogales, whether considered alone or in conjunction with Utter or what was known in the art at the time of the invention, does not teach or suggest data storage elements that together comprise a number of contexts for storing sequences of data. Claim 13 recites data storage elements that together compose a number of contexts for storing information related to a sequence of data to be exchanged through a communications medium connected to the communication controller.

The Office action dated September 8, 2004 contends that this recitation is disclosed by Nogales at column 6, lines 47-57. Presumably then, the Office action contends then that the reference to PCI Interface Logic (column 6, line 34 of Nogales) teaches this concept. See Office action, page 5, section 15. In this citation, Nogales teaches a logic block (PCI Interface Logic (28)) that facilitates the movement of command control blocks (CCBs) from one memory (namely, host memory (13) to another memory (namely, buffer memory (30).

This concept of moving CCBs (which are not an outbound descriptor blocks (ODBs)) between memories is not the same as manipulating an ODB from one data structure to another. Even if one were to assume that these are related concepts, simply citing "PCI Interface Logic" still does not teach data storage elements that together compose a number of contexts for storing information related to a sequence of data to be exchanged through a communications medium connected to the communication controller. Clearly, Nogales shows no understanding or appreciation of representing states and transitions between states that a context representing a task currently executed by the hardware controller can occupy. At best, Nogales teaches PCI Interface Logic that helps facilitate communication between a PCI bus, *i.e.*, host memory and buffer memory by already knowing what transitions are to take place by defined logic. There is certainly no disclosure in Nogales as to how this communication is carried out. It is quite a stretch to argue that data storage elements that together compose a number of contexts for storing information related to a sequence of data to be exchanged through a communications medium connected to the communication controller as recited in claim 13 is taught by the word "logic" when used in conjunction with a fibre channel controller.

C5. Nogales, whether considered alone or in conjunction with Utter or any other known art at the time of the invention, does not teach or suggest a number of logical managers having associated data structures. Claim 13 recites logical managers that are each associated with a data structure and that each carries out operations on behalf of contexts stored within the associated data structure.

In the rejections of claims 1 and 8, the Office action admits, "Nogales fails to explicitly teach having a number of managers containing related/common states." See the Office action dated September 8, 2004, section 5, page 3. However, in the rejection of claim 13, the Office action contends that Nogales does teach this concept in citing column 6, lines 29-34. Further, the Office action originally stated that this recitation was well known in the art at the time of the invention. In an Advisory action dated December 15, 2004, the Examiner cited the reference Utter as an example of this known art. See section 10 of the Advisory action citing column 7,

lines 50-63 of Utter. Clearly, a misunderstanding of what the present invention entails is prevalent throughout the prosecution. Simply put Utter does not cure the deficiency admitted by the Office action and Nogales is still deficient in its teachings.

Furthermore, Nogales falls significantly short of teaching all of the recitations of claim 13. In essence, the Examiner holds the position that Nogales, by disclosing that the GLM block performs parallel-to-serial conversions or vice versa and that grouping together similar tasks as a generally accepted and known programming tenet renders obvious each and every recitation in claim 13. Such broad, conclusory statements do not come close to adequately addressing the issue of motivation to combine, are not evidence of obviousness, and therefore are improper as a matter of law. *In re Dembiczak*, 175 F.3d 994, 999, 50 USPQ2d 1614, 1617 (Fed. Cir. 1999).

C6. In view of the foregoing, the Appellants respectfully submit that the rejections of claims 13-18 must fail for impropriety, and that Board must accordingly overturn these rejections.

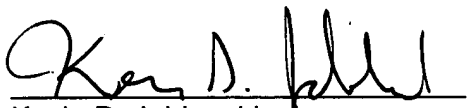
CONCLUSION

The Appellants respectfully consider this application to be in condition for allowance and respectfully requests the Board to overturn the final rejection and that the Examiner pass this application to allowance.

Dated this 4th day of March, 2004.

Respectfully submitted,

Johl et al. (Appellants)



Kevin D. Jablonski
Attorney and Agent for Appellant
Registration No. 50,401
Telephone: (425) 822-3668

APPENDIX: CLAIMS AS PENDING

1. A method for implementing a hardware controller that concurrently executes a number of tasks by carrying out operations on behalf of the tasks, the method comprising:

determining a format for a context, comprising stored information related to a task, that represents the task;

determining possible states, and transitions between states, that a context representing a task currently executed by the hardware controller can occupy at each point in the execution of the task, transitions representing operations performed on behalf of a task by the hardware controller;

partitioning the states and operations carried out by the hardware controller into a number of managers each containing a number of related states and carrying out a number of operations;

associating each manager with a data structure for storing contexts occupying states contained by the manager;

defining a data-structure-manipulator manager that implements the data structures and that transfers contexts from one data structure to another;

defining a command interface to the data-structure-manipulator manager for each manager; and

implementing the managers and data-structure-manipulator manager, according to the determined states and transitions, so that, when a first manager carries out an operation that results in transition of a context to a state contained in a second manager, the first manager generates a command to the data-structure-manipulator

manager to transfer the context from the data structure associated with the first manager to the data structure associated with the second manager.

2. The method of claim 1 wherein tasks are provided to the hardware controller via a signal interface, wherein the hardware controller generates output signals and output data as a result of execution of a task, and wherein operations carried out by managers can be invoked via a signal interface provided for each manager.

3. The method of claim1 wherein the data-structure-manipulator manager comprises a manipulator logic circuit for each manager, a manipulator logic circuit for a manager together with the command interface defined for the manager composing a manipulator within the data-structure-manipulator manager corresponding to the manager.

4. The method of claim 3 wherein the hardware controller includes a timing circuit that generates clock signals are at regular intervals, the intervals including and following a first clock signal and preceding a next clock signal composing a clock cycle, wherein each manager can issue a single command to the manipulator associated with the manager during a single clock cycle, and wherein each manipulator can receive a number of contexts during a single clock cycle for transfer to the data structure associated with the manager corresponding to the manipulator.

5. The method of claim 4 wherein each manipulator can receive a number of commands from a number of managers during each clock cycle.

6. The method of claim 5 wherein related contexts can be linked to one another to form a chain of related contexts that can be transferred together by transferring the first context of the chain of related contexts.

7. The method of claim 6 applied to an outbound sequence manager functionality of a fibre channel interface controller to implement an outbound sequence manager having contexts that store information supplied to the outbound sequence manager in outbound descriptor blocks and having doubly linked-list data structures associated with each manager for storing contexts occupying states contained in the manager, the outbound sequence manager comprising:

a completion manager associated with a completion doubly linked-list;

a credit manager associated with a timer doubly linked-list;

a transmit manager associated with a transmit doubly linked-list;

an outbound descriptor block manager associated with a free doubly linked-list;

a rogue manager associated with a free doubly linked-list;

a non-fabric daemon manager associated with a non-fabric doubly linked-list; and

a centralized list manager data-structure-manipulator manager that transfers contexts from one doubly linked-list to another in response to commands issued to the centralized list manager by the completion manager, credit manager, transmit manager, outbound descriptor manager, rogue manager, and non-fabric daemon, the centralized

list manager having a timer list manipulator, a free list manipulator, a non-fabric list manipulator, a transmit list manipulator, and a completion list manipulator.

8. A method for implementing a hardware controller that concurrently executes a number of tasks, the method comprising:

representing each task executed by the hardware controller as a context, each context occupying a state determined by the contents of at least one field within the context, a context transitioning from one state to another state when the hardware controller carries out an operation on behalf of the task represented by the context;

partitioning hardware controller operations and associated context states into a number of logical managers;

associating each logical manager with one of a number logical data structures for storing contexts occupying states within the logical manager; and

implementing the logical managers and a data-structure manipulator that contains the contexts, logical data structures, and a command interface through which each logical manager issues commands to direct the data-structure manipulator to transfer a context from the data structure associated with the logical manager to a different data structure.

9. The method of claim 8 wherein timing of the hardware controller is controlled by a clock circuit that generates clock signals that define clock cycles, wherein each logical manager may issue at most one command to the data-structure manipulator during each clock cycle, and wherein the data-structure manipulator concurrently executes

commands issued by the logical managers during a clock cycle by serializing the commands according to a defined precedence ordering of the commands.

10. The method of claim 8 wherein the data structures are chosen for efficient storage and retrieval of contexts according to the operations carried out by one or more logical managers associated with the contexts, the data structures chosen from among well-known data structures employed in software programming, including:

singly linked lists;

doubly linked lists;

first-in-first-out queues;

first-in-last-out queues;

stacks;

Graphs;

acyclic graphs, such as binary trees;

arrays;

circular queues; and

combinations of the well-known data structures.

11. The method of claim 8 wherein each logical manager is associated with a signal interface for input and output of signals, wherein operations carried out by logical managers are invoked by signals received through the signal interfaces, and wherein the hardware controller receives tasks and control signals and output data and control signals though a hardware controller interface.

12. The method of claim 11 wherein the hardware controller receives a task via the hardware controller interface and executes the task by:

storing information related to the task within the hardware controller and initializing a context to represent the task;

adding the context to the data structure;

carrying out operations on behalf of the context by the logical manager associated with the data structure in which the context is located, and, when carrying out an operation by a first logical manager results in transition of the context to a state in a second logical manager associated with a different data structure than the data structure in which the context is located, issuing a command from the first logical manager to the data-structure manipulator to transfer the context to the different data structure; and

when all operations that need to be carried out by the hardware controller to execute the task are carried out, generating output data and output signals corresponding to completion of the task by the hardware controller and freeing the context for representing a subsequently received task.

13. A subcomponent controller within a communication controller comprising:

data storage elements that together compose a number of contexts for storing information related to a sequence of data to be exchanged through a communications medium connected to the communication controller;

logical managers that are each associated with a data structure and that each carries out operations on behalf of contexts stored within the associated data structure; and

a data-structure manipulator that implements a number of data structures for storing contexts and that transfer contexts between data structures in response to receiving transfer commands from the logical managers.

14. The subcomponent controller of claim 13 wherein the subcomponent interfaces with external subcomponent controllers via a signal interface and wherein the subcomponent controller receives timing signals at regular intervals that define clock cycles.

15. The subcomponent controller of claim 14 wherein each logical manager may issue a single context transfer command during a single clock cycle, wherein the data-structure manipulator can concurrently receive and carry out one transfer command received from each logical manager during a single clock cycle, wherein the data-structure manipulator serializes all commands received during a single clock cycle by carrying out the commands logically in a predetermined precedence order.

16. The subcomponent controller of claim 13 wherein the communications controller is a fibre channel interface controller and wherein the communications medium is a fibre channel communications medium.

17. The subcomponent controller of claim 16 wherein the subcomponent controller is an outbound sequence manager that receives outbound descriptor blocks from an external subcomponent, stores information related to an outbound sequence and represents a received outbound sequence with a context, and that provides fibre channel frames to an external subcomponent for transmission to the fibre channel medium as a result of executing a task corresponding to a received outbound descriptor block.

18. The subcomponent controller of claim 17 wherein the data structures are doubly linked lists of contexts in which each context may reference a single linked list of related contexts.

19. The subcomponent controller of claim 18 wherein the logical managers include:

a completion manager associated with a completion list;

a credit manager associated with a timer list;

a transmit manager associated with a transmit list;

an outbound descriptor block manager associated with a free list;

a rogue manager associated with a free list;

a non-fabric daemon manager associated with a non-fabric list; and

a centralized list manager that serves as the data-structure manipulator to transfer contexts between lists.

20. The subcomponent controller of claim 19 wherein:

the completion manager can issue commands to the centralized list manager to transfer a context from the completion list to the free list, to transfer a first context from the completion list to the free list and dechain related contexts from the first context and transfer the related contexts to the transmit list, to transfer a first context from the completion list to the free list and dechain related contexts from the first context and transfer related contexts to the completion list, and to transfer a first context from the completion list to the free list and to dechain related contexts from the first context and transfer related contexts to the non-fabric list;

the credit manager can issue commands to the centralized list manager to transfer a context from the timer list to the transmit list, non-fabric list, or the completion list;

the transmit manager can issue commands to the centralized list manager to transfer a context from the transmit list to the timer list or the completion list;

the outbound descriptor block manager can issue commands to the centralized list manager to transfer a context from the free list to the transmit list of the non-fabric list; and

the non-fabric daemon can issue commands to the centralized list manager to transfer a context from the non-fabric list to the transmit list.

PTOL-413A (08-03)
Approved for use through 07/31/2006. OMB 0651-0031
U.S. Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE

Applicant Initiated Interview Request Form

Application No.: 10/842,270 First Named Applicant: M. Grier
Examiner: M. Yigdall Art Unit: 2122 Status of Application: Pending

Tentative Participants:
(1) Michael Yigdall (2) Kevin Jablonski

(3) _____ (4) _____

Proposed Date of Interview: 3/8/2005 **Proposed Time:** 1 PM **(AM/PM)**

Type of Interview Requested:
(1) ☑ **Telephonic** (2) ☐ **Personal** (3) ☐ **Video Conference**

Exhibit To Be Shown or Demonstrated: ☐ **YES** ☑ **NO**
If yes, provide brief description: _____

Issues To Be Discussed

Issues (Rej., Obj., etc)	Claims/ Fig. #s	Prior Art	Discussed	Agreed	Not Agreed
(1) 102(e)	Claims 1-15	Hammond	[ ]	[ ]	[ ]
(2) 102(e)	Claims 16-31	Hammond	[ ]	[ ]	[ ]
(3) 102(e)	Claims 32-38	Saboff/Hammond	[ ]	[ ]	[ ]
(4)			[ ]	[ ]	[ ]

[ ] Continuation Sheet Attached

Brief Description of Arguments to be Presented:

The Prior art does not disclose each and every element as presented in the claims. See attached Proposed Amendment.

An interview was conducted on the above-identified application on _____.

NOTE:
This form should be completed by applicant and submitted to the examiner in advance of the interview (see MPEP § 713.01).
This application will not be delayed from issue because of applicant's failure to submit a written record of this interview. Therefore, applicant is advised to file a statement of the substance of this interview (37 CFR 1.133(b)) as soon as possible.

_____ _____
(Applicant/Applicant's Representative Signature) (Examiner/SPE Signature)